**7 Scikit-Learn Best Practices For Data Scientists**

**Tips for taking full advantage of this machine learning package**

Scikit Learn is one of the go-to libraries for machine learning and it is easy to see why. The package is composed of simple yet effective tools that are explained with very thorough documentation.

However, despite its ease of use, it is easy to make mistakes if you don’t follow certain practices, especially if you are a beginner. I for one fight the urge to facepalm when I see some of the glaring mistakes made using the module in my previous works.

Ultimately, even if you follow the documentation closely, it is still easy to mistakenly omit certain key features or make suboptimal decisions.

Thus, I draw upon my past experiences and delve into 7 Scikit Learn best practices for effectively carrying out predictive data analysis.

**1. Use Scikit Learn (and not Pandas) for feature engineering**

Scikit Learn is designed for machine learning tasks, which of course includes feature engineering. However, it is common for some to use Pandas for certain operations (e.g., one hot encoding) since that is the package most come to learn first.

While the Pandas library is excellent for conducting exploratory data analysis, it can not compare to Scikit Learn in the machine learning space.

The transformers in Scikit Learn are designed for machine learning applications. They can prepare training and testing sets efficiently while avoiding data leakage (if done properly).

Shoehorning Pandas functions into a data pipeline with other Scikit Learn tools will inevitably lead to inefficient procedures that are prone to error.

It is better to instead mainly rely on Scikit Learn for operations pertaining to feature engineering.

**2. Use stratified splits in classification tasks**

Classification tasks can be challenging when the data of interest exhibits data imbalance, where one or more classes are underrepresented.

Fortunately, with stratification, users can maintain the presence of all classes in every subset of the original data.

When splitting the dataset into train and test sets, users can use the stratify parameter.

When splitting the training data into multiple folds for cross-validation, users can use the [StratifiedKFold](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html) class.

**3. Speed up hyperparameter tuning with the n\_jobs parameter**

Hyperparameter tuning can be one of the more time-consuming parts of the data modeling phase. Evaluating multiple combinations of hyperparameters one at a time will naturally be a slow process.

Fortunately, users can speed up hyperparameter tuning methods like grid search and random search by leveraging the n\_jobs parameter, which determines the number of jobs to run in parallel. By default, the n\_jobs value is set to 1, but users can attain results much faster by assigning n\_jobs to -1, which runs jobs parallelly with the use of all available processors.

**4. Assign a random\_state value to attain reproducible results**

A number of feature engineering procedures and machine learning algorithms incorporate randomness. However, a program that utilizes pure randomness will be unable to reproduce its results, which makes it difficult to conduct experiments.

Users can attain reproducible results by setting a seed to the random number generator. For Scikit Learn tools, this is done by assigning a value to the random\_state parameter to an object when applicable. This ensures that the program will yield reproducible results.

Users can set a random\_state value when performing operations such as:

* splitting a dataset into training and testing sets
* configuring a machine learning classifier object
* hyperparameter tuning

Note: The number assigned to the random\_state parameter doesn’t really matter as long as it isn’t changed during the experimentation.

**5. Specify the scoring parameter in hyperparameter tuning**

Hyperparameter tuning methods evaluate models with different combinations of hyperparameters. The purpose of this technique is to identify the hyperparameters that yield peak performance.

However, how can models perform well when they are evaluated with the wrong metric? This is a very possible outcome when you use the default value for the scoring parameter in the Scikit Learn module’s GridSearchCV and RandomizedSearchCV objects.

By default, the grid search and random search evaluates hyperparameters of classification models by accuracy. Unfortunately, this is rarely the suitable metric for a machine learning application.

To avoid this, identify the most fitting evaluation metric for the model of interest and assign it to the scoring parameter. One can find the list of available metrics in the package’s [documentation](https://scikit-learn.org/stable/modules/model_evaluation.html).

If none of the provided metrics are suitable, one can also create their own custom metric with the [make\_scorer](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.make_scorer.html) function. This is a useful feature when users favor one type of error over another.

**6. Transform data with pipelines**

Beginners starting out with Scikit Learn might be used to carrying one transformation at a time. This approach entails using the fit and transform methods multiple times on the training and testing set.

Transforming data in this manner requires many lines of code and can easily lead to mistakes (e.g., using fit on the testing set). So, you will be glad to know that Scikit Learn offers a tool that facilitates these operations with much greater ease: the pipeline.

The Scikit Learn pipeline is a tool that chains together a series of transformations and estimators, enabling users to execute operations with code that is easier to write, read, and debug.

I advocate the use of pipelines constantly and will do so again here; they are just that good.

As an example, we can use a pipeline object to carry out the previous operations:

Compared to the previous snippet, the code here is much more readable and makes it easy to understand all of the steps in the workflow.

Furthermore, all the transformations and modeling in the pipeline object can be executed on a training set with just a single fit method. Moreover, the same transformations can be applied prior to generating predictions from the testing set using a single predict method.

**7. Get familiar with other packages compatible with Scikit Learn**

In the end, the Scikit Learn package’s wide range of tools can not account for every case imaginable.

Thus, it is worth it to familiarize oneself with other packages that are compatible with Scikit Learn. These packages contain tools that can be used together with Scikit Learn for feature engineering and data modeling.

Two noteworthy examples are the [feature\_engine](https://feature-engine.readthedocs.io/en/latest/) and [XGBoost](https://xgboost.readthedocs.io/en/stable/python/index.html) packages, which boast their own unique transformers and machine learning algorithms that can be used with other Scikit Learn tools.

**Conclusion**

The Scikit Learn package has tremendous utility and can be used to tackle all kinds of machine learning problems.

However, users must learn to follow certain best practices to reap the most benefits from the package.

If you’ve found this short overview helpful, you may also want to consider reading these as well:

**[K-Fold Cross Validation: Are You Doing It Right?](https://towardsdatascience.com/k-fold-cross-validation-are-you-doing-it-right-e98cdf3e6690" \t "_blank)**

**[Discussing proper (and improper) ways to perform k-fold cross-validation on datasets](https://towardsdatascience.com/k-fold-cross-validation-are-you-doing-it-right-e98cdf3e6690" \t "_blank)**

[towardsdatascience.com](https://towardsdatascience.com/k-fold-cross-validation-are-you-doing-it-right-e98cdf3e6690" \t "_blank)

**[Why You Should Use Scikit-Learn Pipelines](https://towardsdatascience.com/why-you-should-use-scikit-learn-pipelines-8754b4d1e375" \t "_blank)**

**[This tool takes your code to the next level](https://towardsdatascience.com/why-you-should-use-scikit-learn-pipelines-8754b4d1e375" \t "_blank)**

[towardsdatascience.com](https://towardsdatascience.com/why-you-should-use-scikit-learn-pipelines-8754b4d1e375" \t "_blank)

**[Harnessing Randomness in Machine Learning](https://towardsdatascience.com/harnessing-randomness-in-machine-learning-59e26e82fdfc" \t "_blank)**

**[How “random” should random be?](https://towardsdatascience.com/harnessing-randomness-in-machine-learning-59e26e82fdfc" \t "_blank)**

[towardsdatascience.com](https://towardsdatascience.com/harnessing-randomness-in-machine-learning-59e26e82fdfc" \t "_blank)

I wish you the best of luck in your data science endeavors!